

**IN THE SPECIFICATION**

Please amend the abstract as follows:

A method and system for storing and modifying register transfer language (RTL) described logic types. Upon a declaration of a signal interconnect, a language extension of a register transfer language is defined for the signal interconnect based on the signal [[interconnect"s]] interconnect's type. The language extensions allow different signal interconnect types, such as those used with field programmable gate arrays (FPGA) and standard cells, to be stored in a same file array hierarchy. This storage facilitates changing logic types, thus ultimately resulting in an integrated circuit (IC) that is either smaller (using more standard cells) or more flexible (using more FPGA cells). The transition from one RTL type to another is performed within the physical design cycle, in which wiring, timing and placement of components (information) is performed before masking out the final chip design.

Please amend paragraphs [0005] – [0007] as follows:

[0005] Two types of devices that can implement logic are FPGA (Field Programmable Gate Arrays) and Standard Cell. [[FPGAs]] FPGA's use a 2-dimensional array of logic cells that are programmable, such that the FPGA functions as a custom integrated circuit (IC) that is modified by program code. Thus, a same FPGA can be alternately programmed to selectively perform the function of many different logic circuits. Typically, the programming of the FPGA is persistent until re-programmed at a later time. The persistent nature may be permanent (e.g., by blowing fuses in gates) or modifiable (by storing the programming code in a programmable memory). Standard cell, on the other hand, is hard-wired logic that is not modifiable after it is manufactured. Although it does not have the flexibility of a FPGA, standard cells is usually much faster than FPGA. Furthermore, [[FPGAs]] FPGA's typically have many more gates and logic components than standard cells, since only a part of the FPGA circuit is typically used in any selected programmed configuration. Thus, [[FPGA"s]] FPGA's provide flexibility through

their modifiable nature, but standard cell is faster and takes up less die space to implement a given logic function.

[0006] RTL synthesis takes an RTL file and maps it into a technology supplied by the semiconductor vendor. For example, standard cell synthesis takes a standard cell RTL file and maps out a selection of logic available from the [[vendor"s]] vendor's library, which includes elements such as adders, exclusive [[OR"s (XOR"s)]] OR's (XOR's), AND gates, etc. Similarly, FPGA synthesis maps an FPGA file into an FPGA fabric, supplying program information required for a particular FPGA using configuration files supplied by the vendor.

[0007] In the prior art, FPGA and standard cell logic are created by synthesizing separate RTL files for the FPGA logic and other files for standard cell logic. In some situations, the logic designer may elect to move logic across the boundary, for example from FPGA to standard cell. However, the RTL descriptor files as used in the prior art require FPGA files and standard cell files to be in separate files, as shown in FIG. 1a as FPGA file array 110 and standard cell file array 112. When the logic designer wishes to repartition logic, such as changing an FPGA logic to a standard cell logic, the logic designer must manually remove an RTL FPGA descriptor from the FPGA file array 110, and then manually add an RTL standard cell descriptor for the changed logic to the standard cell file array 112. Although changing logic from FPGA to standard cell or vice versa is fairly simple if only a few functions are involved, when modifying many functions, the process of keeping files organized becomes very difficult and prone to introducing logic errors. This complexity is compounded by the prior art [[RTL"s]] RTL's requirement that FPGA and standard cell files be in different arrays as described above. Thus, it would be beneficial if both FPGA and standard cell files could be stored and manipulated in the same file array, and the boundary between FPGA and standard cell could be dynamically and easily modified during the design process.

Please amend paragraph [0009] as follows:

[0009] Upon a declaration of a signal interconnect, a language extension of a register transfer language is defined for the signal interconnect based on the signal [[interconnect"s]]

interconnect's type. The language extensions allow different signal interconnect types, such as those targeted to field programmable gate arrays (FPGA) and standard cells, to be stored in a same RTL file. This storage facilitates changing logic types, thus ultimately resulting in an integrated circuit (IC) that is either smaller (using more standard cells) or more flexible (using more FPGA cells). Repartitioning of the RTL can be performed within the physical design cycle, in which wiring, timing and placement of components (information) is performed before masking out the final chip design.